AMENDMENTS TO THE CLAIMS

Please amend the claims as indicated below. The language being added is underlined ("__") and the language being deleted contains either a strikethrough ("——") or is enclosed by double brackets ("[[ ]]").

LISTING OF CLAIMS

1. (Currently Amended) A method of monitoring and controlling instruction dependency for microprocessors, the method comprising:

fetching an instruction at thread control element from an instruction buffer;

comparing, with a comparator, one or more source operand identifications of the instruction to one or more temporary register identifications,

comparing, with the comparator, a thread control ID associated with the thread control element with pipeline thread control IDs in a pipeline,

wherein each thread control element and comparator forms a bi-directional correspondence, and wherein each of the one or more temporary register identifications is stored in a temporary register identification pipeline storage location of a set of one or more temporary register identification pipeline storage locations;

verifying whether any of the one or more source operand identifications at the thread control element matches any of the one or more temporary register identifications and verifying whether the thread control ID matches any pipeline thread control IDs; and

in response to a match [[of]] between the source operand identification and the temporary register identification and a match between the thread control ID and a pipeline thread control ID, prohibiting the instruction held in the corresponding thread

control element from executing in that clock cycle, wherein the match corresponds to instruction dependency.

2. (Previously Presented) The method of claim 1, wherein none of the one or more source operand identifications in the thread control element matches any of the one or more temporary register identifications.

3. (Original) The method of claim 2, further comprising the step of initiating execution of the instruction.

4. (Original) The method of claim 3, further comprising the step of verifying whether a destination operand of the instruction is a temporary register.

5. (Original) The method of claim 4, wherein the destination operand is not a temporary register.

6. (Previously Presented) The method of claim 5, further comprising the step of writing a null value into a first pipeline storage location of the set of one or more temporary register pipeline storage locations.

7. (Original) The method of claim 4, wherein the destination operand is a temporary register.

8. (Previously Presented) The method of claim 7, further comprising the step of writing an identification corresponding to the destination operand into a first

pipeline storage location of the set of one or more temporary register pipeline storage locations.

9. (Previously Presented) The method of claim 1, wherein the content in all except the last of the set of one or more temporary register pipeline storage locations is shifted to the next pipeline storage location at the beginning of each clock cycle.

10. (Previously Presented) The method of claim 9, wherein the content of the last pipeline storage location of the set of one or more temporary register pipelines storage locations is released at the beginning of each clock cycle.

11. (Previously Presented) The method of claim 1, wherein at least one of the one or more source operand at the thread control element matches one of the one or more temporary register identifications.

12. (Original) The method of claim 11, further comprising the step of prohibiting execution of the instruction.

13. (Previously Presented) The method of claim 12, further comprising the step of comparing the one or more source operand identifications at the thread control element to the one or more temporary register identifications at the beginning of each clock cycle until none of the one or more source operand identifications matches any of the one or more temporary register identifications.

14. (Original) The method of claim 13, further comprising the step of verifying whether a destination operand of the instruction is a temporary register.

15. (Original) The method of claim 14, wherein the destination operand is not a temporary register.

16. (Previously Presented) The method of claim 15, further comprising the step of writing a null value into a first pipeline storage location of the set of one or more temporary register pipeline storage locations.

17. (Original) The method of claim 14, wherein the destination operand is a temporary register.

18. (Previously Presented) The method of claim 17, further comprising the step of writing an identification corresponding to the destination operand into a first pipeline storage location of the set of one or more temporary register pipeline storage locations.

19. (Currently Amended) A method of monitoring and controlling instruction dependency for microprocessor systems, the method comprising:

a) fetching an instruction at a thread control element;

b) receiving an instruction request at an arbiter, wherein the instruction request is issued from the thread control element;

c) comparing one or more source operand identifications of the instruction at the thread control element to one or more temporary register identifications and comparing a thread control ID associated with the thread control element with pipeline thread control IDs in a pipeline, wherein each of the one or more temporary register identifications is stored in a temporary register identification pipeline storage location of a set of one or more temporary register identification pipeline storage locations, and wherein said one or more source operand instructions at said thread control element is not part of a pipeline or pipelines;

d) verifying whether any of the one or more source operand identifications matches any of the one or more temporary register identifications and verifying whether the thread control ID matches a pipeline thread control ID;

e) in response to a match of the source operand identification and the temporary register identification and a match between the thread control ID and the pipeline thread control ID, prohibiting the instruction held in the corresponding thread control element from executing in that clock cycle, wherein the match corresponds to instruction dependency;

f) if none of the one or more source operand identifications matches any of the one or more temporary register identifications:

f1) verifying whether a destination operand of the instruction is a temporary register; and

f2) if the destination operand of the instruction is a temporary register:

writing an identification corresponding to the destination operand into a first pipeline storage location of the set of one or more temporary register pipeline storage locations;

f3)if the destination operand of the instruction is not a temporary register: writing a null value into a first pipeline storage location of the set of one or more temporary register pipeline storage locations.

20. (Original) The method of claim 19, further comprising the step of initiating execution of the instruction.

21. (Canceled)

22. (Previously Presented) The method of claim 19, if at least one of the one or more source operand identifications at the thread control element matches one of the one or more temporary register identifications in step f), further comprising the steps of:

prohibiting the execution of the instruction;

reiterating step d) until none of the one or more source operand identifications matches any of the one or more temporary register identifications; and

verifying whether a destination operand of the instruction is a temporary register.

23. (Original) The method of claim 22, wherein the destination operand is a temporary register.

24. (Previously Presented) The method of claim 23, further comprising the step of writing an identification corresponding to the destination operand into a first pipeline storage location of the set of one or more temporary register pipeline storage locations.

25. (Original) The method of Claim 22, wherein the destination operand is not a temporary register.

26. (Previously Presented) The method of Claim 25, further comprising the step of writing a null value into a first pipeline storage location of the set of one or more temporary register pipeline storage locations.

27. (Previously Presented) the method of claim 19, wherein the content in all except the last of the set of one or more temporary register pipeline storage locations is shifted to the next pipeline storage location at the beginning of each clock cycle.

28. (Previously Presented) The method of claim 27, wherein the content of the last pipeline storage location of the set of one or more temporary register pipeline storage locations is released at the beginning of each clock cycle.

29. (Previously Presented) A system for instruction dependency monitor and control, comprising:

a set of one or more thread control elements for fetching instructions, wherein said thread control elements are not part of a pipeline stage or pipeline storage location;

a set of one or more comparing elements, wherein each of the one or more comparing elements is directly coupled to a corresponding thread control element in the set of one or more thread control elements; and

a set of one or more temporary register identification pipeline storage locations, wherein the one or more temporary register identification pipeline storage locations are directly coupled to the one or more comparing elements.

30. (Original) The system of claim 29, further comprising an instruction buffer coupled to the one or more thread control elements.

31. (Previously Presented) The system of claim 30, further comprising an arbiter, wherein the arbiter is coupled to the one or more thread control elements, the one or more comparing elements, and the one or more temporary register identification pipeline storage locations.

32. (Original) The system of claim 31, further comprising an arithmetic logic unit (ALU) coupled to the arbiter.

33. (Original) The system of claim 32, further comprising a set of one or more input data buffers coupled to the arbiter, wherein each input data buffer corresponds to a thread control element of the one or more thread control elements.

34. (Previously Presented) The system of claim 33, further comprising a set of one or more temporary register buffers coupled to the arbiter, wherein each temporary register buffer corresponds to a thread control element of the one or more thread control elements.

35. (Previously Presented) A system for instruction dependency monitor and control, comprising:

a set of one or more thread control elements for fetching instructions wherein said thread control elements are not part of a pipeline stage or storage location;

a set of one or more comparing elements, wherein each of the one or more comparing elements is directly coupled to a corresponding thread control element in the set of one or more thread control elements and wherein each thread control element and comparing element forms a bi-directional correspondence;

a set of one or more temporary register identification pipeline storage locations, wherein the one or more temporary register pipeline storage locations are directly coupled to the one or more comparing elements, and

an arbiter coupled to the thread control elements, the comparing elements, and the temporary register pipeline storage locations in each stage of a pipeline or pipelines.